BioCASE: Accelerating software development of genome-wide filtering applications

Rosana Montes, María M. Abad-Grau [*]

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Granada, Granada 18071, Spain
{rosana,mabad}@ugr.es

Abstract. Due to the high pace that algorithms for scanning genome-wide datasets are produced, most of the software is quickly released. As a consequence, they usually lack in system portability and only provide a text-based user interface. BioCASE is an open-source tool developed to assist bioinformaticians in the task of producing software to perform any kind of computationally-expensive genome-wide scanning. BioCASE will produce efficient software with some other added features such as to be portable, to have a graphical user interface (GUI) and to be easily set up. A first version of BioCASE (http://bios.ugr.es/biocase) has been completed and it is being currently used in the design of a genome-wide tool to perform variable selection of genotype data sets based on multivariate association and transmission-disequilibrium tests.

Key words: genome, CASE tools, feature selection, filtering

1 Introduction

Commercially-available genome-wide chip arrays are quickly increasing the number of genotype markers along the chromosomes they provide for a sample of individuals. Nowadays arrays with more than 300 thousand simple nucleotide polymorphisms (SNPs) markers (binary variables) are currently used. Projects meant to find genotypes in association with complex diseases are the main inquirers of these arrays. To reduce costs, these genome-wide genotype data sets must first be filtered by using different algorithms for feature selection. It is only after the in-silico filtering is completed that molecular biologists, geneticists and any other bio-researchers perform molecular-based experimentation.

Algorithms for genome filtering or for some required preprocessing step such as haplotype estimation are developed at a very high rate. Regarding hardware, the long computational times due to hardware limitations the bio-researchers may suffer, is one of the main issues for those algorithms to be broadly used.

[*] Corresponding author: María del Mar Abad Grau, Lenguajes y Sistemas Informáticos, ETS Ingeniería Informática y de Telecomunicaciones, Universidad de Granada, c/ Periodista Daniel Saucedo Aranda s/n, 18071 Granada, Spain

However in this work our main concern was about software. Regarding software, the bottle-neck in their use is mainly produced because of a deficient application of principles in software engineering. Most of genome-wide filtering algorithms are usually developed by highly specialized researchers either in bioinformatics or in biostatistics. Thus, they usually provide very simple software with their algorithms, which is platform-dependent and only has a text-based user interface. From the developer point of view, the main goal is to provide software that use new filtering algorithms or some preprocessing tasks, visualize genome-wide maps after the algorithms have been applied or both. The software is released to be used in the same platform where the algorithms were tested. The source code –if provided– may be highly difficult to compile in a different platform and it usually lacks a graphical user interface (GUI). From the end-user point of view, bio-researchers, the software may turn out to be difficult to install, it may require a CPU model they do not have and/or an operative system they do not feel comfortable with and it may demand from them to use a command-line interpreter instead of a friendly GUI.

Perhaps the most widely used tool to assist software development for computational biology and bioinformatics (CBB) is Bioconductor [1]. This software is written in R, a high-level interpreted language for statistical computing, and thus it is not computationally feasible when it comes about any kind of genome-wide algorithms [1]. There are substantial issues in the way applications implementing these time-consuming algorithms were built, some examples including platform-dependence as in GEAR (a tool for analyzing DNA copy number changes [2]), text user interface as in BLink (a tool for computing bimarker linkage disequilibrium) [3] or both as in fastPhase [4] (a tool for haplotype estimation). On top of this, bio-researchers quite often have to deal with more than one tool to perform a task. As en example BMapBuilder [5] builds chromosome-wide linkage-disequilibrium (LD) maps, it is portable and it incorporates a GUI. However, for those maps to be built from genotype data sets, other software to first compute pairwise LD must be used. To the best of our knowledge, there are no specific open-source CASE tools to assist software development of applications performing genome-wide search.

Section 2 describes the main functionality of BioCASE, together with details about specific coding, deployment approaches and external packages BioCASE can use. Design methods used by BioCASE are referred in Section 3. Section 4 shows an example of how BioCASE can be used to build a tool for both genome-wide filtering and visualization. Conclusions and future aims are exposed in Section 5.

2 Functionality

BioCASE is a Computer-Aided Software Engineering (CASE) tool for the development and deployment of genome-wide filtering applications that meets many conceptual and computational challenges. Our methodology and design allows for independent and parallel development of code for bioinformatics. With the creation of extensible software we promote the achievement of better capabilities in research results given that investigators can explore and interact with data resources and algorithms. Some of the features the software to be developed with BioCASE is expected to have are: (1) efficiency, (2) user friendly, (3) portability, (4) editions for different natural languages and (5) reproducibility, use and extension. For those features to be easily provided, BioCASE requires the following functionality: (1) Cross-compilation of GNU c++ code for different platforms, (2) use of a metalanguage for user interface specification and high-level tasks, (3) resources for an efficient management of different natural languages and (4) distribution of open-source code under the General Public License (GPL) v2.0.

3 Design

BioCASE is fully implemented in Java, a multiplatform object-oriented language which enables the use of abstraction, interface definition and inheritance. Java permits to quickly design. It also provides file system data management, numerical capabilities, attractive widgets, flexible visualization (in 2D and 3D), access to databases and portability. Moreover, our java code is able to interact with cross-compiled c++ code for computationally-expensive genome-wide scanning.

Distributed development is enabled in BioCASE. It requires the use of collaborative tools that permits to work simultaneously on the same component. With a Subversion repository (svn) shared code and other resources such as data and documentation are accessible to all members of the project. We adopted SVN versioning system as it is based on a secure protocol (SSH).

The BioCASE project is designed as a set of *tools* that resolve different issues and facilitates the development of genome-wide filtering applications. It packs many tools in the same environment with a uniform fashion and use. To enable an efficient definition and use of actual and future tools we use external libraries: (1) The *Locale* and *ResourceBundle* classes from J2SE API allow to separate language specific tags in text properties files. Our application will permit to easily develop software in different languages. (2) The JDOM Project (`http://www.jdom.org/`) is an open-source library which enables us to use XML configuration files to implement two important properties of BioCASE. We specify modularized tasks from the model in a configuration XML file and we are able to define the GUI (menus, windows and wizards) of the software to be developed with BioCASE.

The BioCASE tool is hosted at `http://bios.ugr.es/biocase/` where the user can find documentation, notifications, data and the latest release. The project deployment is based on the JavaTM Archive (JAR) file format due the fact that it enables you to bundle multiple files (binary class files, images, configuration data and locale resources) into a single archive file. Portable source code will be released under GPL v2.0.

Documentation is presented in our system in two formats: as complete manuals in Adobe's portable document format (PDF) and as navigate documents with dynamic content (HTML). While the manuals are downloadable from the web, the hypertext help is accessible directly from the graphical user interface as we have implemented a naive browser inside BioCASE.

4 Using BioCASE

The first software that is being developed by using BioCASE is TDTer, a genome-wide tool for comparative computation of different transmission-disequilibrium tests and graphical visualization of genome-wide TDT-based maps (Figure 1).

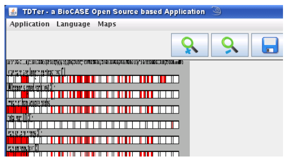

Fig. 1. A snapshot of TDTer.

5 Conclusions

We have presented a multiplatform open-source CASE tool that can be used to assist development of genome-wide filtering applications by using an xml configuration schema for the GUI and tasks definitions. We plan to extend their features so that the development of other biological computational processes and statistical models used in computational genomics can be performed.

Acknowledgments. The authors have been partially supported by the Spanish Research Program under project TIN2007-67418-C03-03, the Andalusian Research Program under project P08-TIC-03717 and by the European Regional Development Fund (ERDF).

References

1. Gentleman, R.C., et al.: Bioconductor: open software development for computational biology and bioinformatics. Genome Biology **5**(5) (2004) 1–16
2. Kim, T.M., Jung, Y.C., Rhyu, M.G., Jung, M.H., Chung, Y.J.: Gear: genomic enrichment analysis of regional dna copy number changes. Bioinformatics **24**(3) (2008) 420–421
3. Sebastiani, P., Abad-Grau, M.M.: Bayesian estimates of linkage disequilibrium. BMC Genetics **8** (2007) 1–13
4. Scheet, P., Stephens, M.: A fast and flexible statistical model for large-scale population genotype. data: Applications to inferring missing genotypes and haplotypic phase. Am J Hum Genet **78** (2006) 629–644
5. Abad-Grau, M.M., Montes, R., Sebastiani, P.: Building chromosome-wide LD maps. Bioinformatics **22**(16) (2006) 1933–1934